(12) UK Patent Application (19) GB (11) 2 325 543 (13) A

(43) Date of A Publication 25.11.1998

(21) Application No 9806473.6

(22) Date of Filing 27.03.1998

(30) Priority Data
(31) 08832430 (32) 02.04.1997 (33) US

(71) Applicant(s)
International Business Machines Corporation
(Incorporated in USA - New York)
Armonk, New York 10504, United States of America

(72) Inventor(s)
Sandeep K Singhal
Gennaro A Cuomo
Binh Q Nguyen
Richard J Redpath

(74) Agent and/or Address for Service
S R Davies
IBM United Kingdom Limited,
Intellectual Property Dept, Hursley Park, Winchester,
Hampshire, SO21 2JN, United Kingdom

(51) INT CL[6]
A63F 9/22 , G06F 19/00

(52) UK CL (Edition P)
G4A AKS AUXS

(56) Documents Cited
EP 0714684 A1 WO 98/32507 A1 WO 95/30465 A1
Microsoft DirectPlay white paper, http://www.microsoft.com/directx/pavilion/dplay/default.asp

(58) Field of Search
UK CL (Edition P) G4A AKS AUXS
INT CL[6] A63F 9/22 , G06F 19/00
Online: WPI, INTERNET, COMPUTER

(54) Abstract Title
Multi-user game system

(57) Disclosed is an object-orientated game management system which enables games developers to design games using a platform-independent object model without needing to concern themselves with the details of multi-user communications management. An object is created (lobby session - 400) which manages the creation of and access to a multiple user game session (406). A server (lobby server - 402) maintains a list of available games and details of those participants currently partaking in those games. From the game session (406) users can either create a game session of their own at a central location (410), whence they simultaneously create a client session for themselves (408), or can join in with pre-existing sessions (414) by invoking client objects (412) to manage their access to that game.

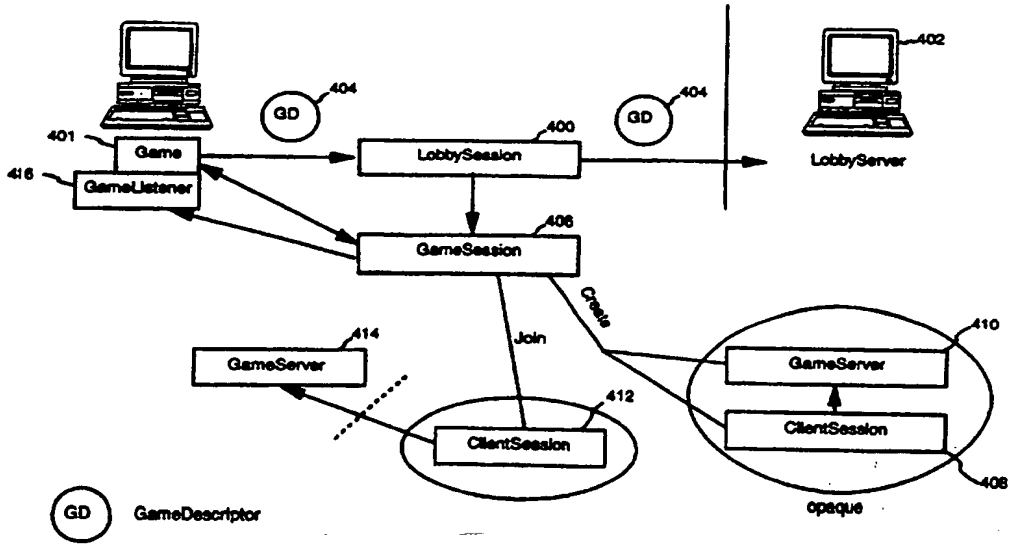

FIG. 4

GB 2 325 543 A

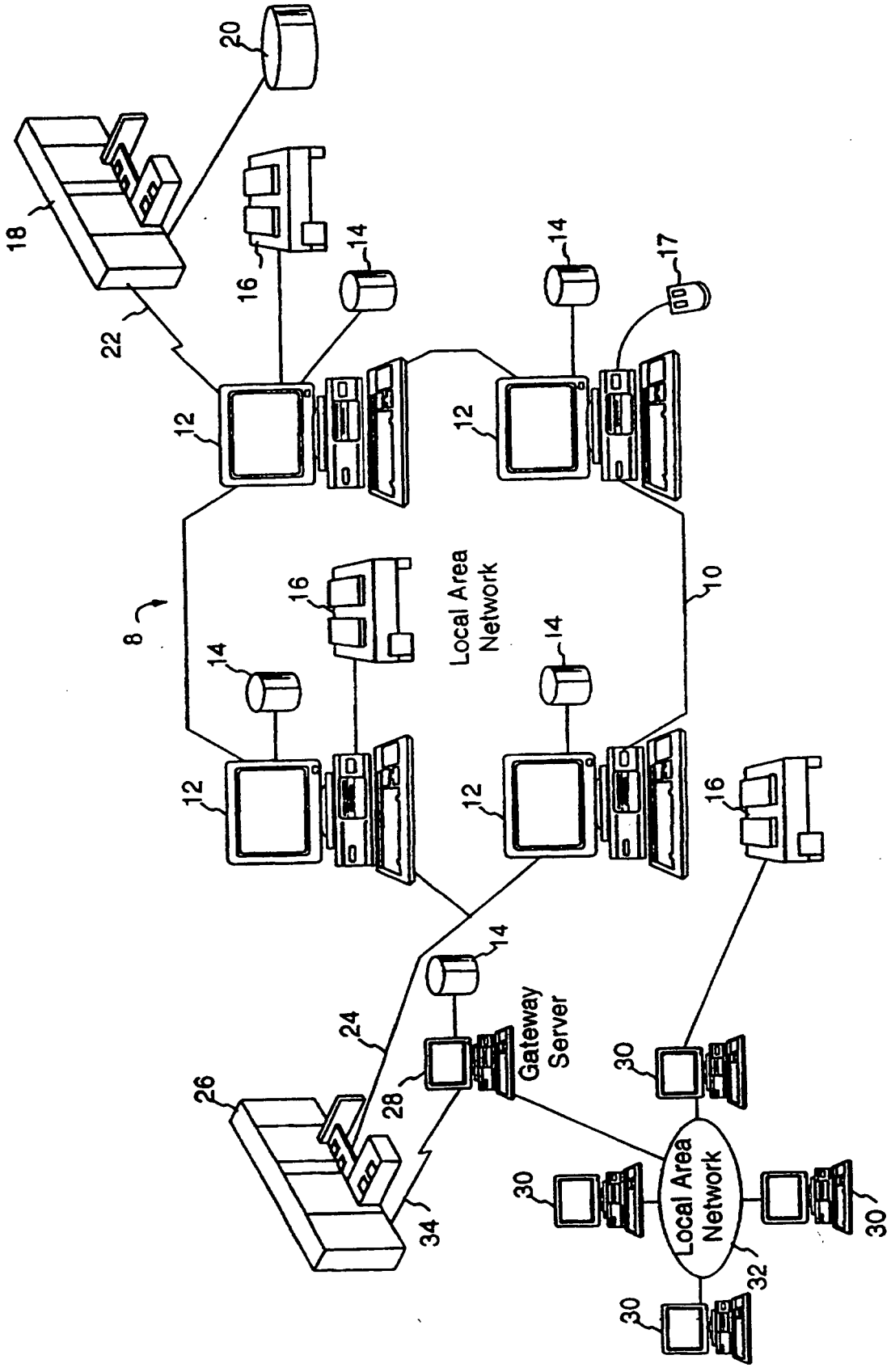

FIG 1

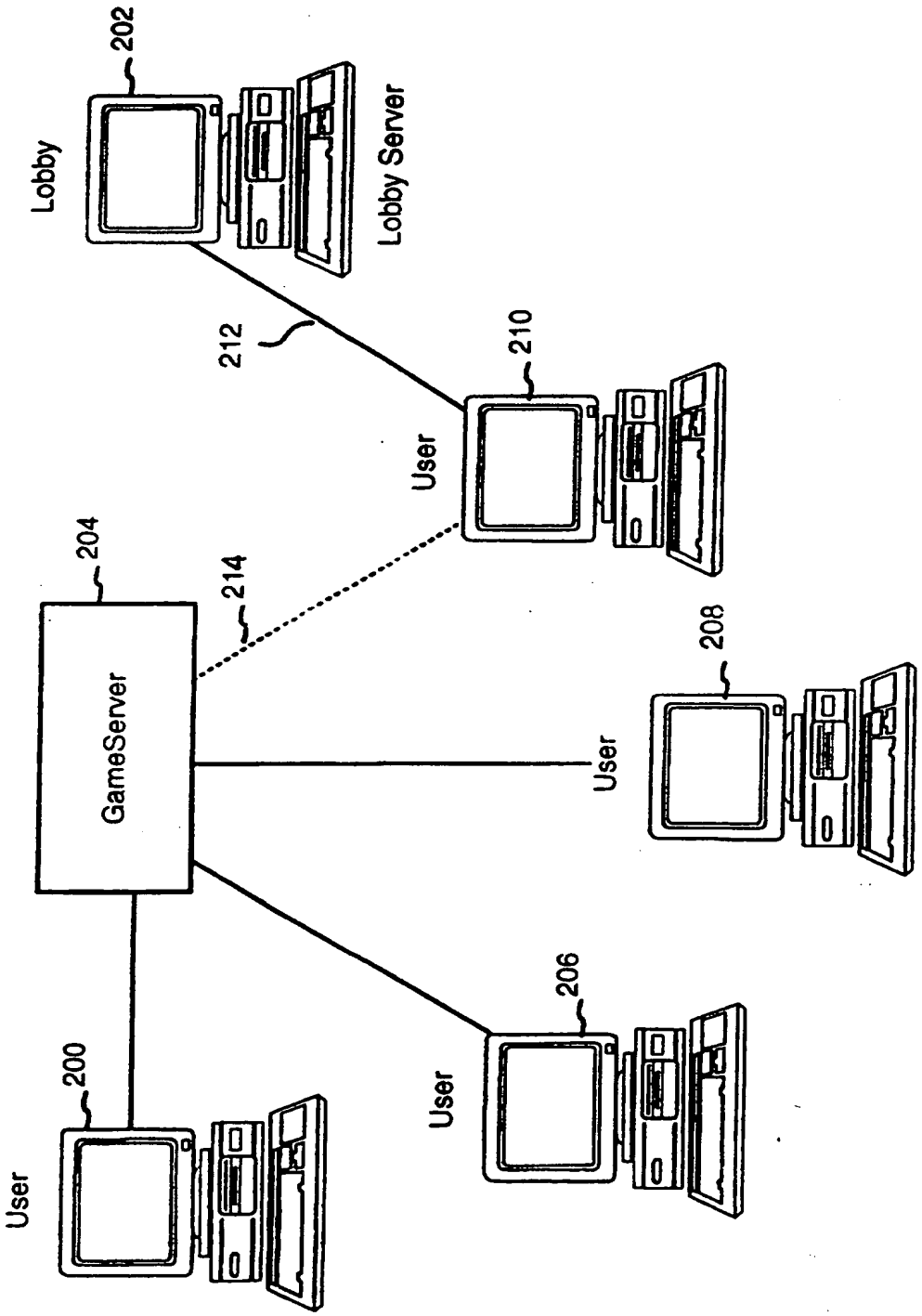

FIG. 2

```
LobbySession
  public abstract boolean register(String Password);
  public abstract void destroy();
  public abstract Vector getGames(String clubs);
  public abstract GameSession joinGame(GameDescriptor gd);
  public abstract void unjoinGame(GameDescriptor gd);
  public abstract Vector LurkGame(GameDescriptor gd);
  public abstract GameSession createGame(GameDescriptor gd);
  public abstract void set HeartbeatListener(Heartbeat5Listener h5l);
```

300

```
GameSession
  public abstract void setGameListener(GameListener gl);
  public abstract void sendText(String text);
  public abstract void sendGameEvent(byte bytes []);
  public abstract void endGame();
  public abstract void quitGame();
  public abstract void forceStart();
  public abstract long currentTime();
```

302

306

```
GameDescriptor
  public String type;.
  public String universalName;
  public String topic;
  public String ownerName;
  public String ownerHostname;
  public String clubs;
  public int  port;
  public int  players;
  public int  level;
  public long  startTime;
```

317

304

```
GameListener
  public abstract void gameStart();
  public abstract void gameStop();
  public abstract void gameAddPlayer(String universalName, String friendlyName);
  public abstract void gameRemovePlayer(String universalName, String friendlyName);
  public abstract void gameEvent(String senderUniversalName, byte [] event Parameters);
  public abstract void gameNewTextMessage(String SenderUniversalName, String text);
```

FIG. 3

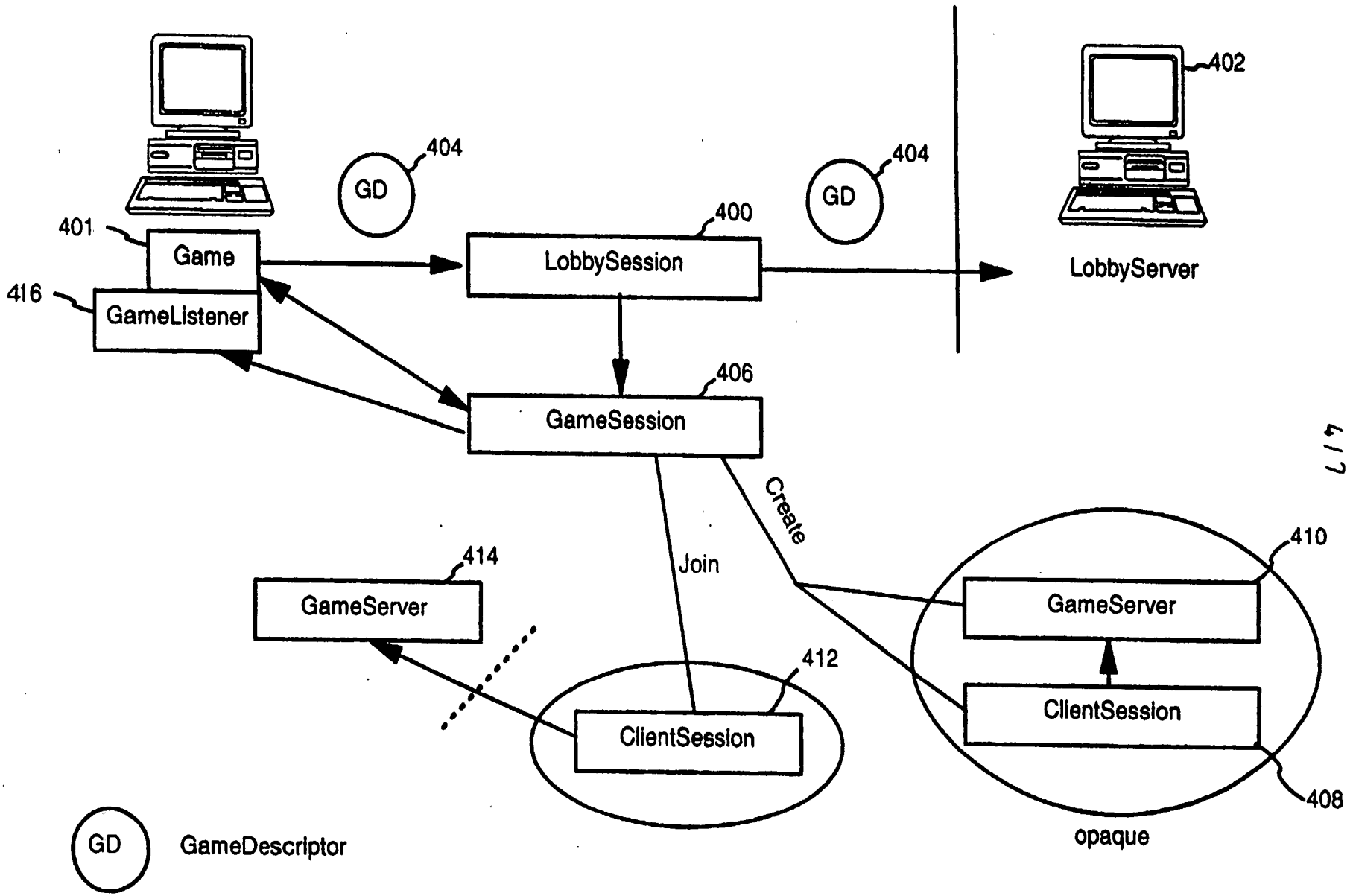

FIG. 4

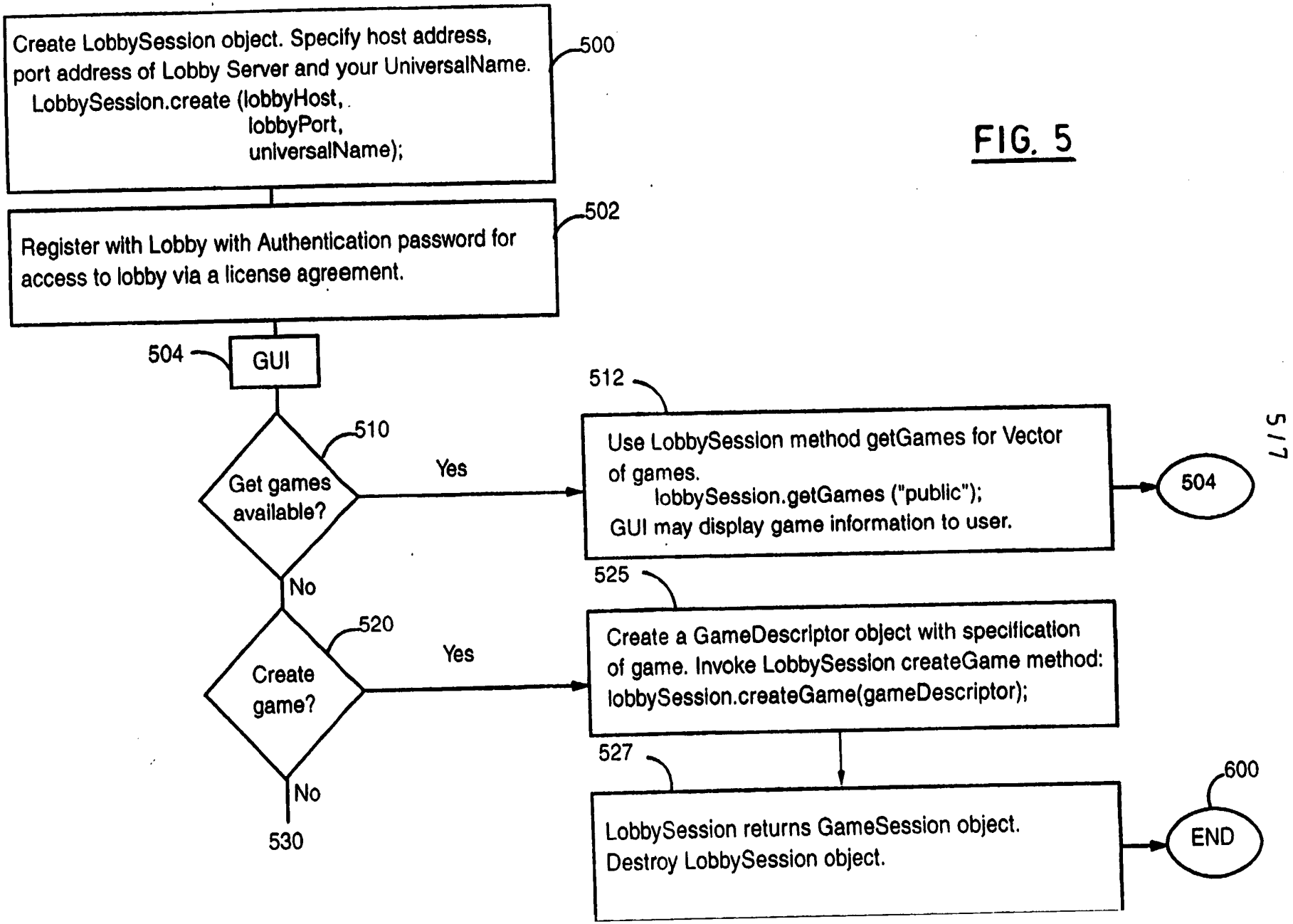

FIG. 5

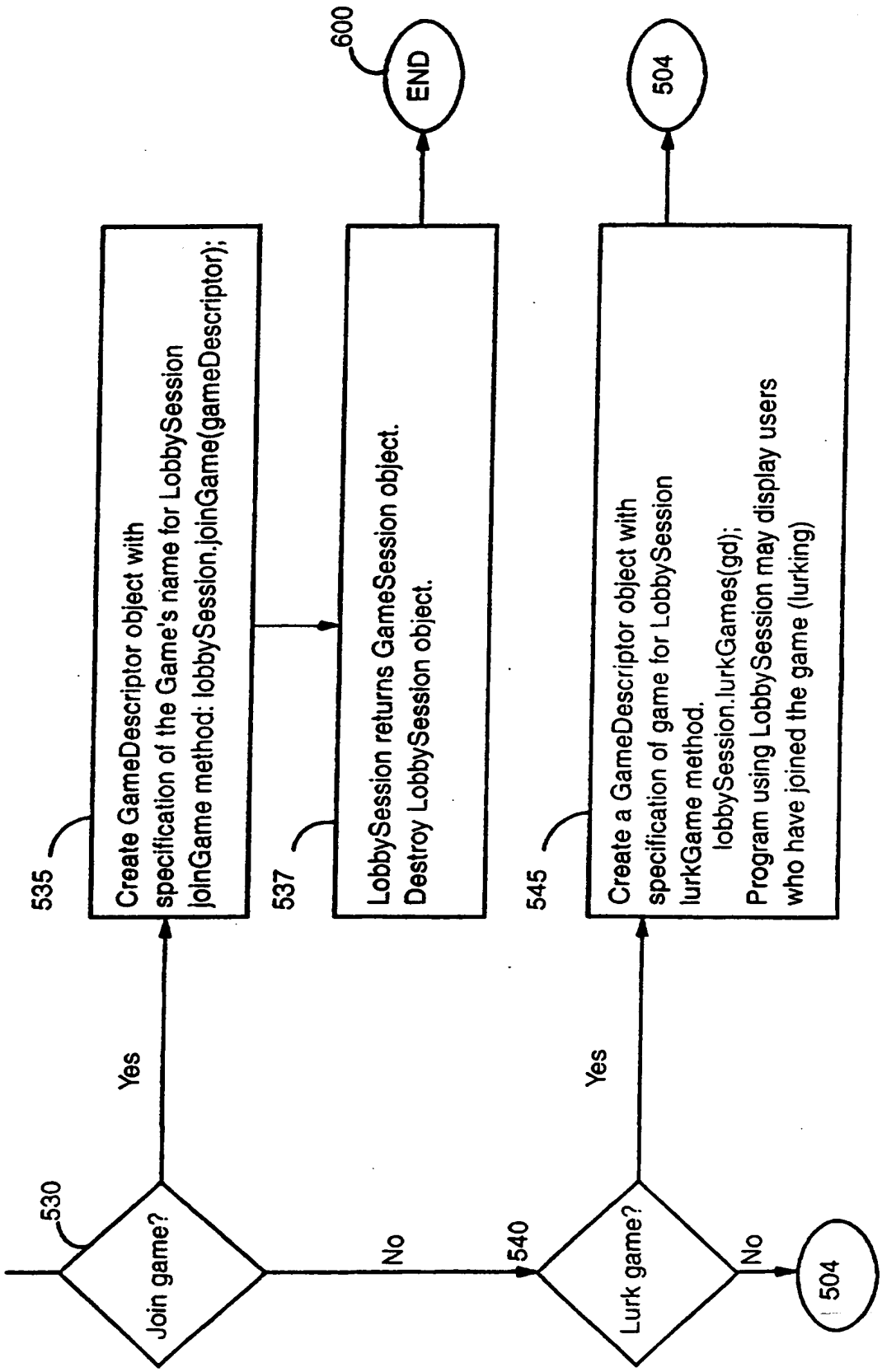

FIG. 6

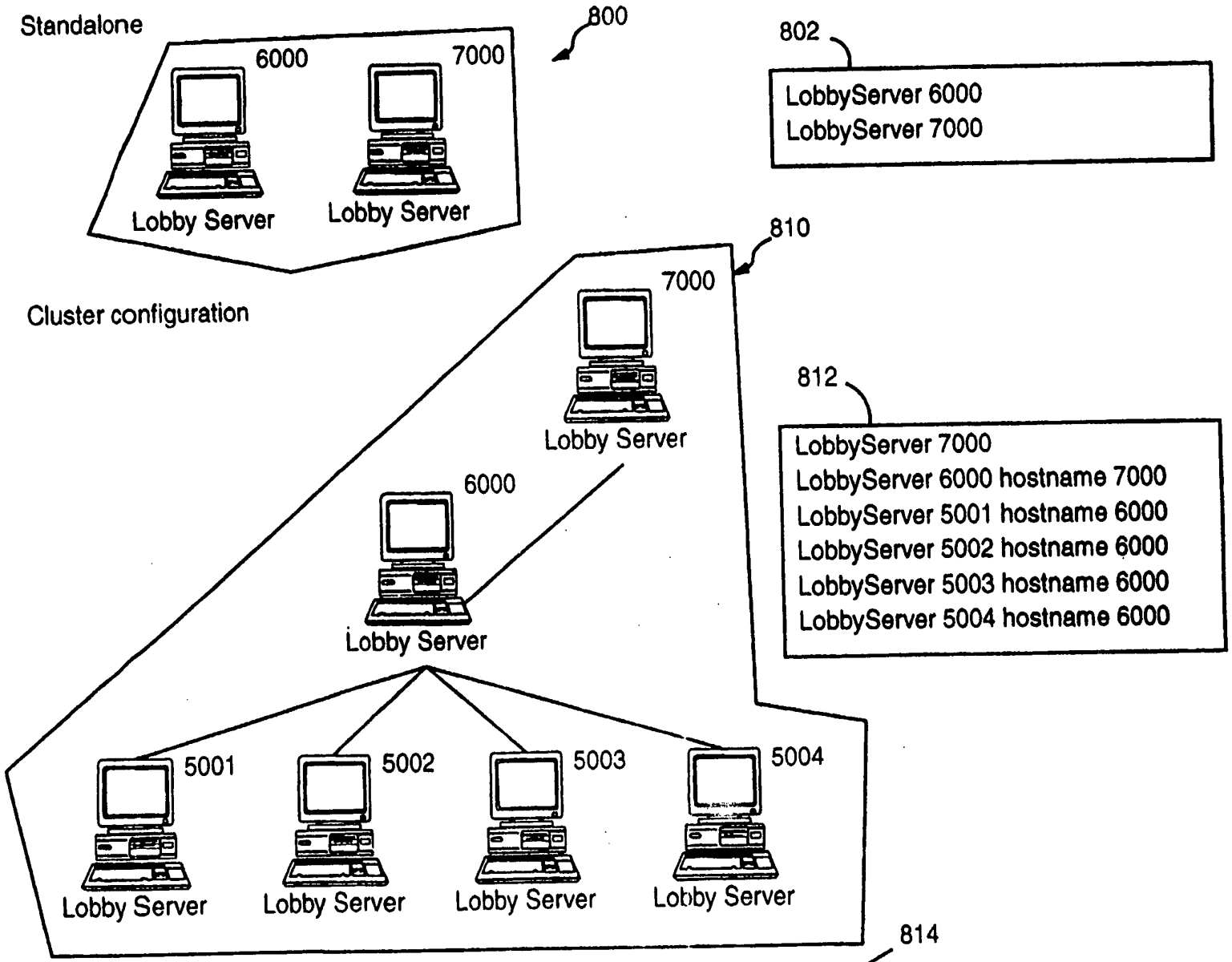

FIG. 7

717

2325543

MULTI-USER GAME SYSTEM AND METHOD

This invention relates to a method and system for providing an interface in a data processing system to a communication process which allows multiple users to connect to or participate in a multi-user game.

Multi-user games such as "Quake", "Populous", and "MUDs" (Multiple User Dungeons) have proliferated using proprietary client-server models. The creators of these games are faced with the task of managing multi-user communication which is typically far outside the primary game content or game developer's expertise. Corporations such as TEN (Total Entertainment Network) are trying to provide a high-level programming interface to achieve this communication goal for game creators. Many such companies are engaged in the competition to provide a simple interface which requires little or no knowledge of multi-user and network programming.

Prior art includes proprietary services that are provided on the Internet by private and publicly available companies such as TEN (Total Entertainment Network) and Microsoft Corporation. TEN's current implementation relies on the game developer to integrate the game with a proprietary lobby implementation to service users and allow multi-user games to be joined. Microsoft Corporation offers DirectPlay which encapsulates a COM object, thereby requiring the developer to run on Microsoft platforms and requiring the use of a Browser that supports ActiveX. The monolithic architecture of ActiveX dictates a complex programming interface.

The overall concept desired is to provide a game lobby in which users can open a game with their name at a central place so that others may see and join that game. The user can set the criteria for starting this game, such as the number of players or start time, whichever comes first.

Accordingly, the invention provides a method of providing a network interface in a data processing system to scalably connect multiple users for a multi-user game, comprising the steps of:

using a LobbySession object to encapsulate access to a Lobby Server; and

using said LobbySession object to retrieve a list of available games.

The step of using said LobbySession object can further create a new game, join an existing game, or lurk an existing game. A preferred embodiment further uses said LobbySession object to obtain a GameSession object, which can then be used to encapsulate creation of a GameServer (typically for game creation), and/or to encapsulate client access to a GameServer (typically for joining a pre-existing game).

In a preferred embodiment, said Lobby Server is configured to run as part of a cluster of Lobby Servers, wherein said LobbySession selects a Lobby Server from among those available in said cluster.

The invention further provides a system for providing a network interface in a data processing system to scalably connect multiple users for a multi-user game, comprising:

means for using a LobbySession object to encapsulate access to a Lobby Server; and

means for using said LobbySession object to retrieve a list of available games.

The invention further provides a computer program product recorded on computer readable medium for providing a network interface in a data processing system to scalably connect multiple users for a multi-user game, comprising:

computer readable means for using a LobbySession object to encapsulate access to a Lobby Server; and

computer readable means for using said LobbySession object to retrieve a list of available games.

It will be appreciated that the system and computer program product of the invention will further benefit from substantially the same set of preferred features as the method of the invention.

Thus a method and system are described herein for providing a game lobby and game session for a game developer in a multi-user environment, using a seamless object model to facilitate any multi-user game. A simple programming interface allows for a multi-user environment, offering scalability without restricting the use of the communication system within certain types of networks. The seamless configurable clustering of servers optimally facilitates scalable user connections to create games and join games via a game lobby.

The platform-independent object model removes the burden of multi user communication management from the game developer. The object model enables the development of game lobby services by multiple vendors and allows games to freely operate with any of these lobby implementations. Using the method and system described herein, game developers can conne users to each other, create/join multi-user games, and enable inter-use communications in a scalable manner.

A preferred embodiment of the invention will now be described in detail by way of example only with reference to the following drawings:

Figure 1 is a pictorial representation of a data processing syste

Figure 2 illustrates a Multi-User Game environment;

Figure 3 illustrates interface support objects for Gaming;

Figure 4 illustrates an example of the underlying architecture of objects in the preferred embodiment;

Figure 5 illustrates a flowchart for facilitating multi-user gamin in the preferred embodiment; and

Figure 6 is a flowchart illustrating the starting of a Lobby Serve in the preferred embodiment.

Referring to Figure 1, there is depicted a graphical representatic of a data processing system 8. As may be seen, data processing system 8 may include a plurality of networks, such as Local Area Networks (LAN) 1 and 32, each of which preferably includes a plurality of individual computers 12 and 30, respectively. Of course, those skilled in the art will appreciate that a plurality of Intelligent Work Stations (IWS) coupled to a host processor may be utilized for each such network. Each said network may also consist of a plurality of processors coupled via a communications medium, such as shared memory, shared storage, or an interconnection network. As is common in such data processing systems, each individual computer may be coupled to a storage device 14 and/or a printer/output device 16 and may be provided with a pointing device such as a mouse 17.

The data processing system 8 may also include multiple mainframe computers, such as mainframe computer 18, which may be preferably couple to LAN 10 by means of communications link 22. The mainframe computer 18 may also be coupled to a storage device 20 which may serve as remote storage for LAN 10. Similarly, LAN 10 may be coupled via communications link 24 through a sub-system control unit/communications controller 26

and communications link 34 to a gateway server 28. The gateway server 28 is preferably an IWS which serves to link LAN 32 to LAN 10.

With respect to LAN 32 and LAN 10, a plurality of documents or resource objects may be stored within storage device 20 and controlled by mainframe computer 18, as resource manager or library service for the resource objects thus stored. Of course, those skilled in the art will appreciate that mainframe computer 18 may be located a great geographic distance from LAN 10 and similarly, LAN 10 may be located a substantial distance from LAN 32. For example, LAN 32 may be located in California while LAN 10 may be located within North Carolina and mainframe computer 18 may be located in New York.

Software program code is typically stored in the memory of a storage device 14 of a stand alone workstation or LAN server from which a developer may access the code for distribution purposes, the software program code may be embodied on any of a variety of known media for use with a data processing system such as a diskette or CD-ROM or may be distributed to users from a memory of one computer system over a network of some type to other computer systems for use by users of such other systems. Such techniques and methods for embodying software code on media and/or distributing software code are well-known and will not be further discussed herein.

Referring to Figure 2, there is depicted a multi-user game environment. A multi-user game has been established by a User 200. The game is registered in a Lobby 202. Other users join the game by connecting to a GameServer 204. Two additional Users 206 and 208 are currently joined to the game created by the User 200 via GameServer 204. A new User 210 contacts the LobbyServer 202, as indicated by line 212, to find out what games are available and to retrieve the network addresses for their respective GameServers, and then the user can select and join the desired game, as indicated by dotted line connection 214 to GameServer 204.

Figure 3 illustrates the interface support objects for the game developer. The objects are: LobbySession 300, GameSession 302, GameListener 304 and GameDescription 306. These four objects encapsulate the underlying process of creating a game, posting the created game to a lobby, and enabling the game to be joined by users. Only four objects are required and thus disclosed to the game developer to facilitate a multi-

user game and entry via a Lobby. This is in contrast to a relatively large API required by the prior art.

Referring to Figure 4, a Game application 401 uses a LobbySession 400 to communicate to a Lobby Server 402 which manages games that are available to join and games that are to be created. The GameDescriptor (GD) 404 is used by all objects for uniformity to disseminate game information. From the LobbySession 400 a GameSession object 406 can be created and returned to the Game application 401 to facilitate communication among the players of a particular game. This GameSession object 406 has an opaque data model which allows support for creating games and joining games. When a game is created, the GameSession object 406 will create a GameServer 410 to manage the game and will also create a ClientSession 408 to connect to that GameServer for joining the game c behalf of the game creator. When a game is joined, the GameSession objec 406 will only create a ClientSession 412 that connects to an existing (and typically remote) GameServer 414 on behalf of the joining user. In other words, joining a game creates a ClientSession 408 and creating a game creates a ClientSession 408 and a GameServer 410. During the Game, application 401 may subsequently control the game by invoking methods on GameSession 406. It receives notifications about game events through th GameListener object 416, supplied by the Game application 401.

Referring to Figure 5, a flow chart is illustrated. For a game developer to support a lobby environment for a multi-user game, a LobbySession object is created at block 500. The game developer must specify the network address or location for the Lobby Server (this Lobby Server network address may be obtained by a variety of well-known method and, therefore, will not be discussed herein). The LobbySession provides a registration method for authentication with the server at block 502. Although not shown in Figure 5, it is to be understood that without a proper authentication, the Lobby services may not be available to that user. A graphical user interface (GUI) is provided to the user at block 504 to allow ease of use. It is then determined at decision block 510 whether the user has made a request to get a list of available games. If the response to decision block 510 is yes, a LobbySession method "getGames" (see Figure 3) is used to obtain a Vector to the games at block 512. Processing returns to the GUI at block 504.

If the response to decision block 510 is no, it is determined at decision block 520 whether or not a user wants to create a game. If the

response to decision block 520 is yes, a GameDescriptor object is created with a game specification and passed to the LobbySession "createGame" method (see Figure 3) at block 525. The LobbySession returns a GameSession object and destroys the LobbySession object previously created at block 500. Processing then ends at 600.

If the response to decision block 520 is no, it is determined at decision block 530 whether or not a user wants to join a game. If the response to decision block 530 is yes, a GameDescriptor object is created with a specification of the Game's name and passed to the LobbySession "joinGame" method (see Figure 3) at block 535. The LobbySession then returns a GameSession object and destroys, at block 537, the LobbySession object previously created at block 500. Processing then ends at 600.

If the response to decision block 530 is no, it is determined at decision block 540 whether or not a user wants to "lurk" a game (meaning that the user does not want to join a game yet, but rather wants to see who is currently registered to play the game). If the response to decision block 540 is yes, a GameDescriptor object is created with a specification of the Game's name for the LobbySession "lurkGame" method (see Figure 3) at block 545, which returns a Vector listing the players currently registered in the specified game. After block 545 or if the response to decision block 540 is no, processing returns to the GUI at block 504.

Referring to Figure 6, the Lobby Server can be started as a plurality of standalone servers, generally identified by reference numeral 800, or in a recursive cluster configuration, generally identified by reference numeral 810, through the sample parameter specifications 802 or 812, respectively. The syntax of this parameter specification is described by a usage message 814. Any Lobby Server that is started can be used by a LobbySession, and clustering and data replication is performed, whether running standalone or running as part of a cluster, transparently. Therefore, the design as taught herein of using four objects (see Figure 3) does not restrict the ability of the process to utilize clustering as opposed to standalone configurations.

CLAIMS

1. A method of providing a network interface in a data processing system to scalably connect multiple users for a multi-user game, comprising the steps of:

using a LobbySession object to encapsulate access to a Lobby Server; and

using said LobbySession object to retrieve a list of available games.

2. The method of Claim 1, further comprising the step of using said LobbySession object to create a new game.

3. The method of Claim 1, further comprising the step of using said LobbySession object to join an existing game.

4. The method of Claim 1, further comprising the step of using said LobbySession object to lurk an existing game.

5. The method of any preceding Claim, further comprising the step of using said LobbySession object to obtain a GameSession object.

6. The method of Claim 5, further comprising the step of using said GameSession object to encapsulate creation of a GameServer.

7. The method of Claim 5 or 6, further comprising the step of using said GameSession object to encapsulate client access to a GameServer.

8. The method of any preceding Claim, wherein said Lobby Server is configured to run as part of a cluster of Lobby Servers.

9. The method of Claim 8, wherein said LobbySession selects a Lobby Server from among those available in said cluster.

10. A system for providing a network interface in a data processing system to scalably connect multiple users for a multi-user game, comprising:

means for using a LobbySession object to encapsulate access to a Lobby Server; and

means for using said LobbySession object to retrieve a list of available games.

11. The system of Claim 10, further comprising means for using said LobbySession object to create a new game.

12. The system of Claim 10, further comprising means for using said LobbySession object to join an existing game.

13. The system of Claim 10, further comprising means for using said LobbySession object to lurk an existing game.

14. The system of any of Claims 10 to 13, further comprising means for using said LobbySession object to obtain a GameSession object.

15. The system of Claim 14, further comprising means for using said GameSession object to encapsulate creation of a GameServer.

16. The system of Claim 14 or 15, further comprising means for using said GameSession object to encapsulate client access to a GameServer.

17. The system of any of Claims 10 to 16, wherein said Lobby Server is configured to run as part of a cluster of Lobby Servers.

18. The system of Claim 17, wherein said LobbySession selects a Lobby Server from among those available in said cluster.

19. A computer program product recorded on computer readable medium for providing a network interface in a data processing system to scalably connect multiple users for a multi-user game, comprising:

computer readable means for using a LobbySession object to encapsulate access to a Lobby Server; and

computer readable means for using said LobbySession object to retrieve a list of available games.

20. The program product of Claim 19, further comprising computer readable means for using said LobbySession object to create a new game.

21. The program product of Claim 19, further comprising computer readable means for using said LobbySession object to join an existing game.

22. The program product of Claim 19, further comprising computer readable means for using said LobbySession object to lurk an existing game.

23. The program product of any of Claims 19 to 22, further comprising computer readable means for using said LobbySession object to obtain a GameSession object.

24. The program product of Claim 23, further comprising computer readable means for using said GameSession object to encapsulate creatio of a GameServer.

25. The program product of Claim 23 or 24, further comprising compute readable means for using said GameSession object to encapsulate client access to a GameServer.

26. The program product of any of Claims 19 to 25, wherein said Lobby Server is configured to run as part of a cluster of Lobby Servers.

27. The program product of Claim 26, wherein said LobbySession selects Lobby Server from among those available in said cluster.

The Patent Office

Application No: GB 9806473.6
Claims searched: 1 to 18

Examiner: Grant Bedford
Date of search: 16 September 1998

Patents Act 1977
Search Report under Section 17

Databases searched:

UK Patent Office collections, including GB, EP, WO & US patent specifications, in:

UK Cl (Ed.P): G4A (AKS AUXS)

Int Cl (Ed.6): A63F 9/22, G06F 19/00

Other: Online: WPI, INTERNET, COMPUTER

Documents considered to be relevant:

Category	Identity of document and relevant passage		Relevant to claims
A	EP 0714684 A1	(NET GAME)	-
X,E	WO 98/32507 A1	(KAON INTERACTIVE) See page 5 paras. 1 and 2, and figure 2 in particular	1 and 10 at least
X	WO 95/30465 A1	(CATAPULT) See abstract and claim 1 in particular.	1 and 10 at least
X,P	Microsoft DirectPlay white paper, http://www.microsoft.com/directx/pavilion/dplay/default.asp		1 and 10 at least

X	Document indicating lack of novelty or inventive step	A	Document indicating technological background and/or state of the art.
Y	Document indicating lack of inventive step if combined with one or more other documents of same category.	P	Document published on or after the declared priority date but before the filing date of this invention.
&	Member of the same patent family	E	Patent document published on or after, but with priority date earlier than, the filing date of this application.